**Министерство науки и высшего образования Российской Федерации** федеральное государственное автономное образовательное учреждение высшего образования

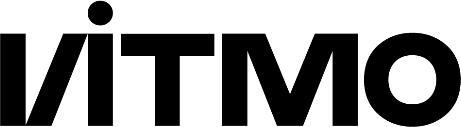
### «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

### Отчет

по лабораторной работе №5 «Процедуры, функции, триггеры в PostgreSQL**»** по дисциплине **«Проектирование и реализация баз данных»**

Автор: Тюленев А.С. Факультет: ИКТ Группа: К3240

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

**Оглавление**

[Вариант 6. БД «Пассажир» 3](#_bookmark0)

[Ход работы 4](#_bookmark1)

[Вывод 12](#_bookmark2)

**Цель работы:** овладеть практическими создания и использования процедур, функций и триггеров в базе данных PostgreSQL.

**Оборудование:** компьютерный класс.

**Программное обеспечение:** СУБД PostgreSQL, pgadmin 4.

### Практическое задание:

1. Создать процедуры/функции согласно индивидуальному заданию (часть 4).

* 1. Модифицировать триггер (триггерную функцию) на проверку корректности входа и выхода сотрудника (см. Практическое задание 1 Лабораторного практикума (Приложение)) с максимальным учетом «узких» мест некорректных данных по входу и выходу).
  2. Создать авторский триггер по варианту индивидуального задания.

**Вариант 6. БД «Пассажир»**

Описание предметной области: Информационная система служит для продажи железнодорожных билетов. Билеты могут продаваться на текущие сутки или предварительно (не более чем за 45 суток). Цена билета при предварительной продаже снижается на 5%. Билет может быть приобретен в кассе или онлайн. Если билет приобретен в кассе, необходимо знать, в какой. Для каждой кассы известны номер и адрес. Кассы могут располагаться в различных населенных пунктах.

Поезда курсируют по расписанию, но могут назначаться дополнительные поезда на заданный период или определенные даты.

По всем промежуточным остановкам на маршруте известны название, тип населенного пункта, время прибытия, отправления, время стоянки.

Необходимо учитывать, что местом посадки и высадки пассажира могут быть промежуточные пункты по маршруту.

БД должна содержать следующий минимальный набор сведений: Номер поезда. Название поезда. Тип поезда. Пункт назначения. Пункт назначения для проданного билета. Номер вагона. Тип вагона. Количество мест в вагоне. Цена билета. Дата отправления. Дата прибытия. Дата прибытия для пункта назначения проданного билета. Время отправления. Номер вагона в поезде. Номер билета. Место. Тип места. Фамилия пассажира. Имя пассажира. Отчество пассажира. Паспортные данные.

**Задание 1.1 (ЛР 1 БД).**  Выполните инфологическое моделирование базы данных системы. (Ограничения задать самостоятельно.)

**Задание 1.2.** Создайте логическую модель БД, используя ИЛМ (задание 1.1).  Используйте необходимые средства поддержки целостности данных в СУБД.

**Задание 2.** Создать запросы:

* Свободные места на все поезда, отправляющиеся с вокзала в течение следующих суток.
* Список пассажиров, отправившихся в Москву всеми рейсами за прошедшие сутки.
* Номера поездов, на которые проданы все билеты на следующие сутки.
* Свободные места в купейные вагоны всех рейсов до Москвы на текущие сутки.
* Выручка от продажи билетов на все поезда за прошедшие сутки.
* Общее количество билетов, проданных по всем направлениям в вагоны типа “СВ”.
* Номера и названия поездов, все вагоны которых были заполнены менее чем наполовину за прошедшие сутки.

**Задание 3.** Создать представление:

* для пассажиров о наличии свободных мест на заданный рейс;
* количество непроданных билетов на все поезда, формирующиеся за прошедшие сутки (номер поезда, тип вагона, количество).

**Задание 4.** Создать хранимые процедуры:

* Для повышения цен в пригородные поезда на 20%.
* Для создания нового рейса на поезд.
* Для формирования общей выручки по продаже билетов за сутки.

**Задание 5.**  Создать необходимые триггеры.

# Ход работы:

1. **Процедура для повышения цен в пригородные поезда на 20%.**

CREATE OR REPLACE PROCEDURE raise\_suburban\_ticket\_prices()

LANGUAGE plpgsql

AS $$

BEGIN

UPDATE ticket

SET ticket\_price = ticket\_price \* 1.20

WHERE trip\_id IN (

SELECT tp.id

FROM trip tp

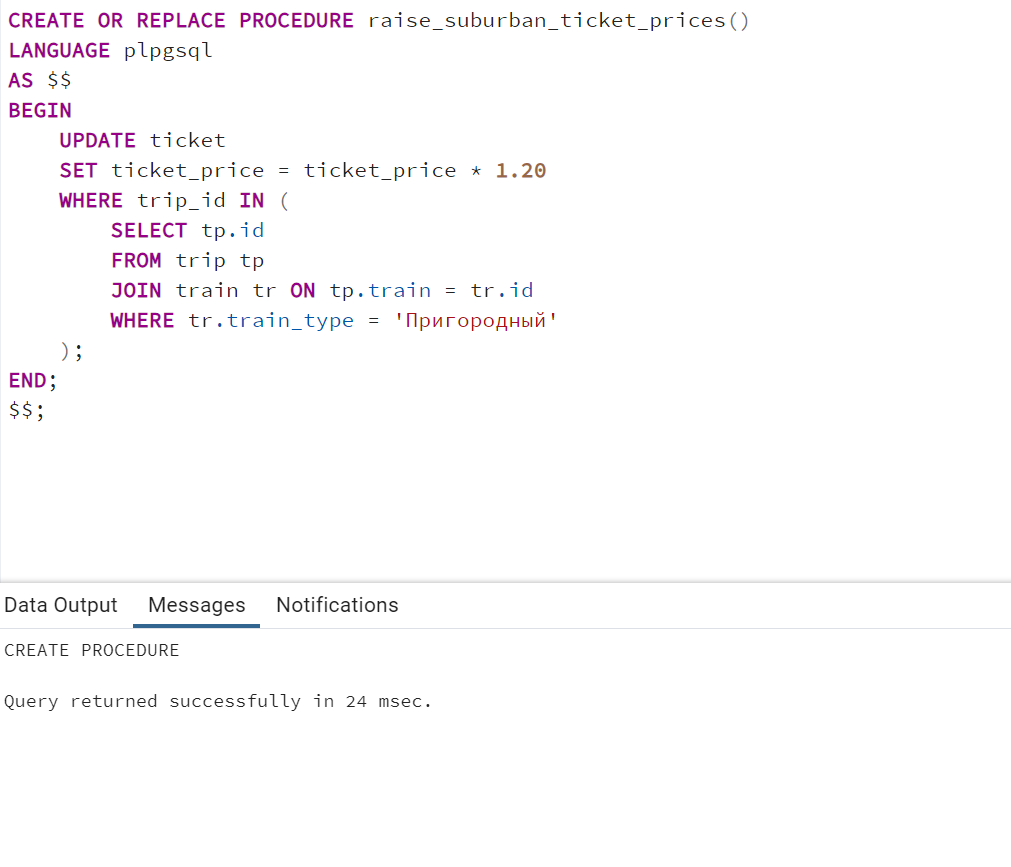
JOIN train tr ON tp.train = tr.id

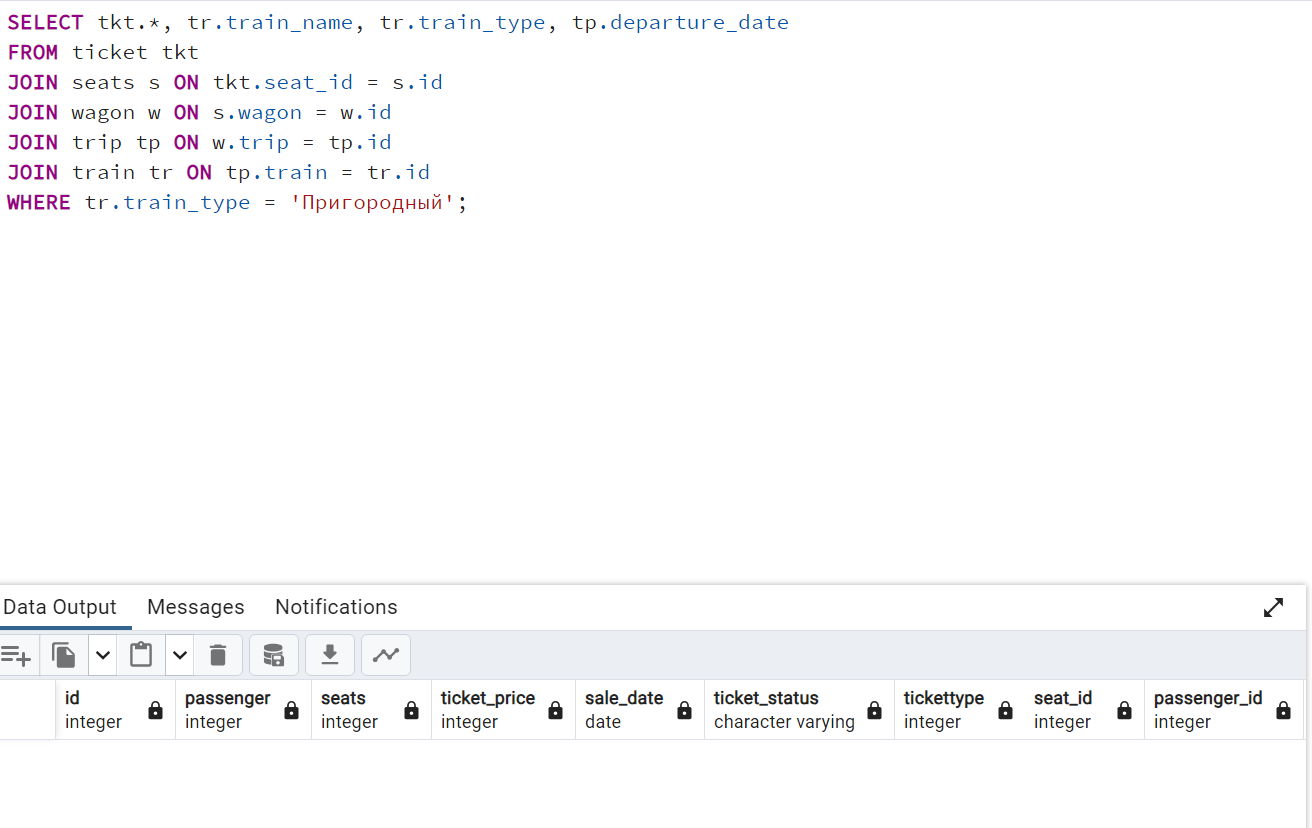
WHERE tr.train\_type = 'Пригородный'

);

END;

$$;

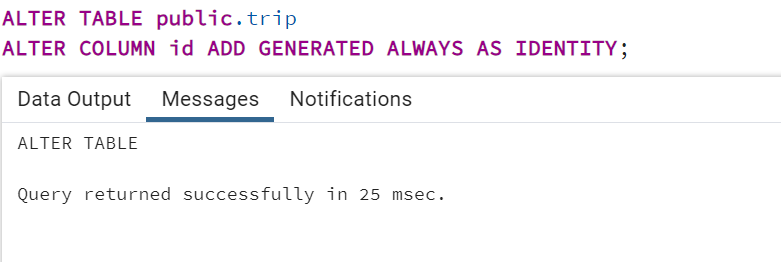




1. **Процедура для создания нового рейса на поезд.**

ALTER TABLE public.trip

ALTER COLUMN id ADD GENERATED ALWAYS AS IDENTITY;



CREATE OR REPLACE PROCEDURE public.create\_new\_train\_trip(

train\_id INTEGER,

p\_departure\_date DATE,

p\_arrival\_date DATE,

p\_status VARCHAR

)

LANGUAGE plpgsql

AS $$

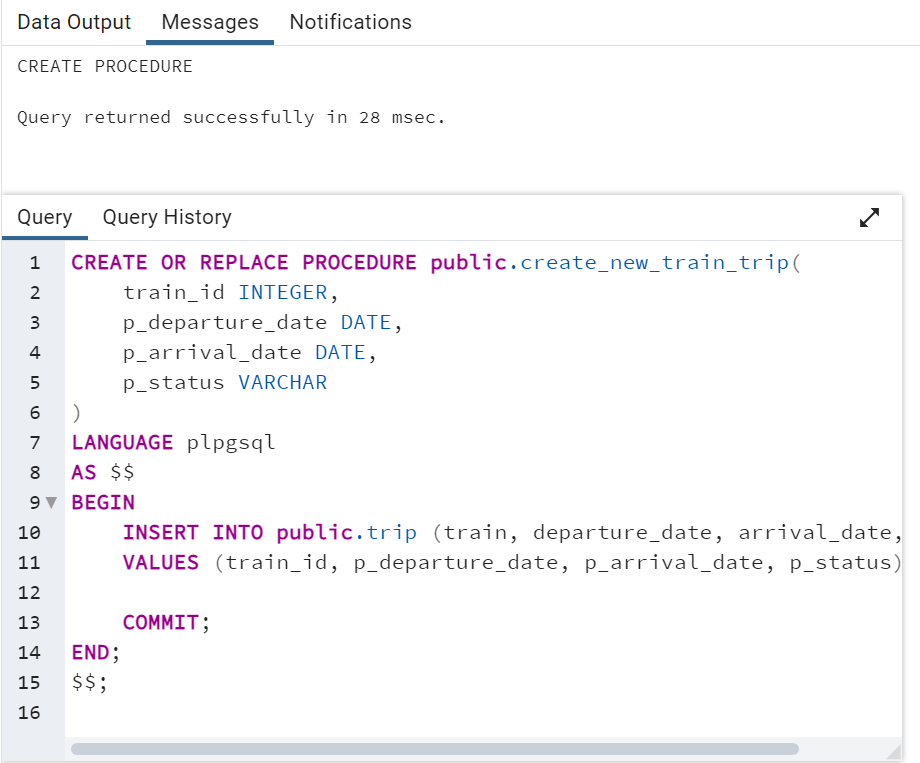
BEGIN

INSERT INTO public.trip (train, departure\_date, arrival\_date, status)

VALUES (train\_id, p\_departure\_date, p\_arrival\_date, p\_status);

COMMIT;

END;

$$;

1. **Процедура для формирования общей выручки по продаже билетов за сутки.**

CREATE OR REPLACE FUNCTION calculate\_daily\_revenue(p\_date DATE)

RETURNS INTEGER AS $$

DECLARE

v\_total\_revenue INTEGER;

BEGIN

SELECT COALESCE(SUM(ticket\_price), 0)

INTO v\_total\_revenue

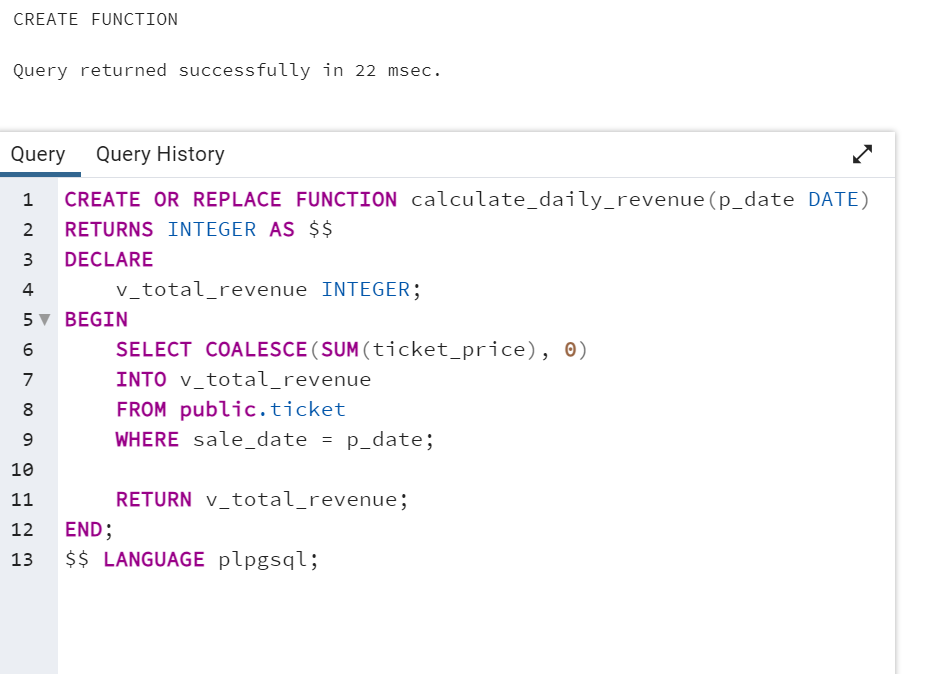
FROM public.ticket

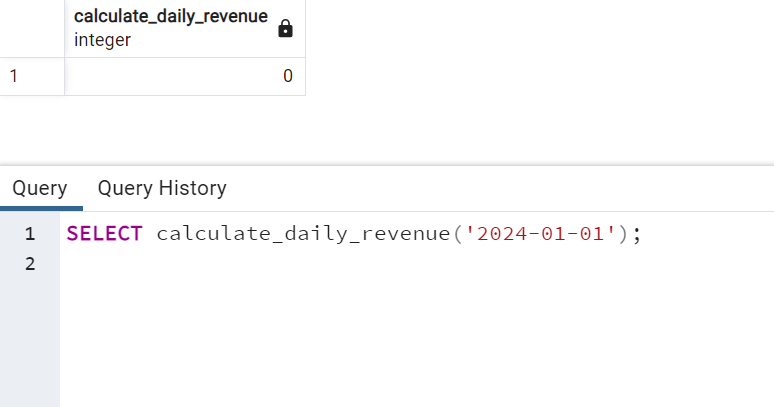
WHERE sale\_date = p\_date;

RETURN v\_total\_revenue;

END;

$$ LANGUAGE plpgsql;





**Триггер**

Триггер, который изменяет статус занятости места, при вставки новой записи в ticket

CREATE OR REPLACE FUNCTION update\_seat\_occupancy()

RETURNS TRIGGER AS $$

BEGIN

IF TG\_OP = 'INSERT' THEN

UPDATE seats

SET occupancy\_status = 'Занято'

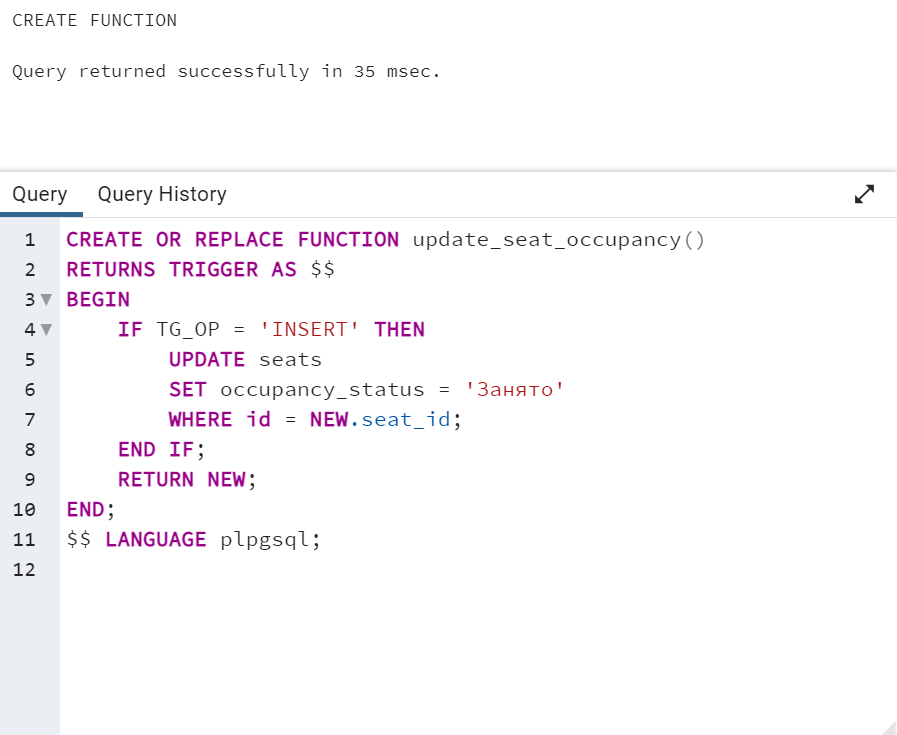
WHERE id = NEW.seat\_id;

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

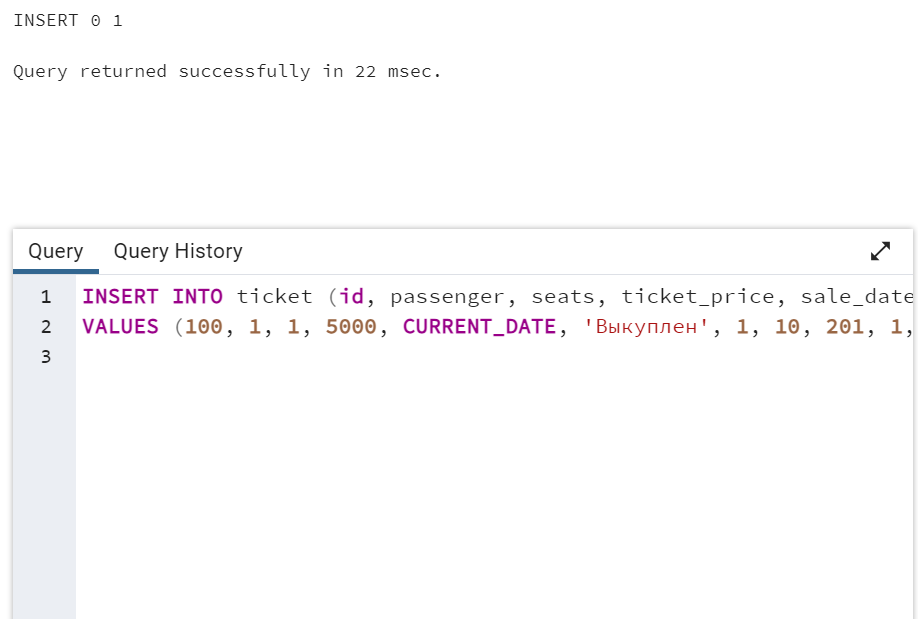


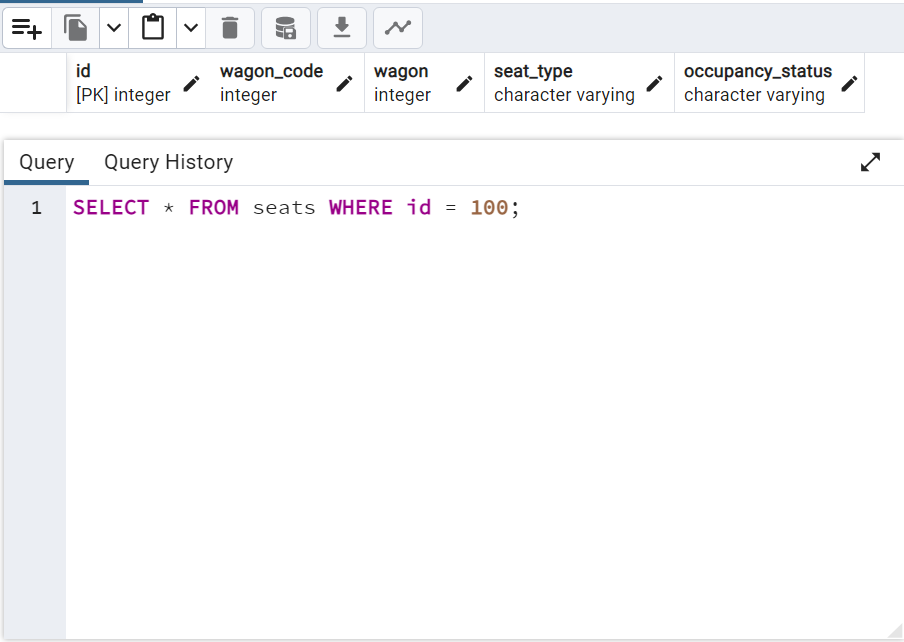
CREATE TRIGGER ticket\_after\_insert

AFTER INSERT ON ticket

FOR EACH ROW EXECUTE FUNCTION update\_seat\_occupancy();







# Вывод

## В процессе выполнения данной лабораторной работы мы освоили навыки создания и использования процедур, функций и триггеров в PostgreSQL.